

Systems Development and Implementation

Lesson 3

Part I RAD Plan Session

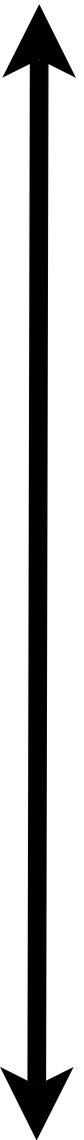
- RAD example
- Requirements analysis
- Teams develop a detailed plan to iteratively prototype their project with RAD in mind
 - Video Game Rentals
 - Opera Management Software
 - Crazy 8 Corporate Campus
- Prepare the plan
- Present the plan
- Critique the plan

RAD Example

- 1
 - Two year project to replace NMS systems
 - Select a COTS product (2 months)
 - Conduct 2 pilot sites (go/no go)
 - Gather additional req'ts from lessons learned
- 2
 - Upgrade pilot sites (custom code, s/w add-ons)
 - Deploy solution to 1/2 the sites
 - Gather additional feedback
- 3
 - Enhance solution further
 - Deploy to remaining sites, upgrade those existing
- 4
 - Maintenance mode. Gather patch requests.
 - Push patches as needed, repair hardware

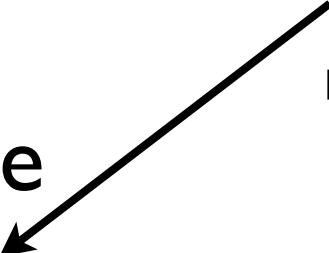
SDLC

Phase		Activity	What you have when you're done
1	Identify	Identify unmet business requirements. (ex JAD)	Business requirements, justification.
2	Initiate	Scope out the system and project plan	Costs, staffing, timetable, critical success factors, locations effected, participants
3	Analyze	Requirements analysis	Comparison of alternative solutions
4	Logical Design	Architecture - blocks of functions - data movement	Functional details about data, inputs, outputs
5	Physical Design	Technical details	Technical specifications about networks, storage, compute power
6	Prototype	Limited field trial with partial functionality. (ex RAD)	Lessons learned about functionality, scalability, usability, reliability, fine-tune implementation schedule.
7	Implement	Real hardware, software	System is fully operational
8	Maintain	Support, repair, enhance	This never ends until the system is obsolete, cannot be upgraded, and is replaced



Requirements Analysis

These are the
business
requirements

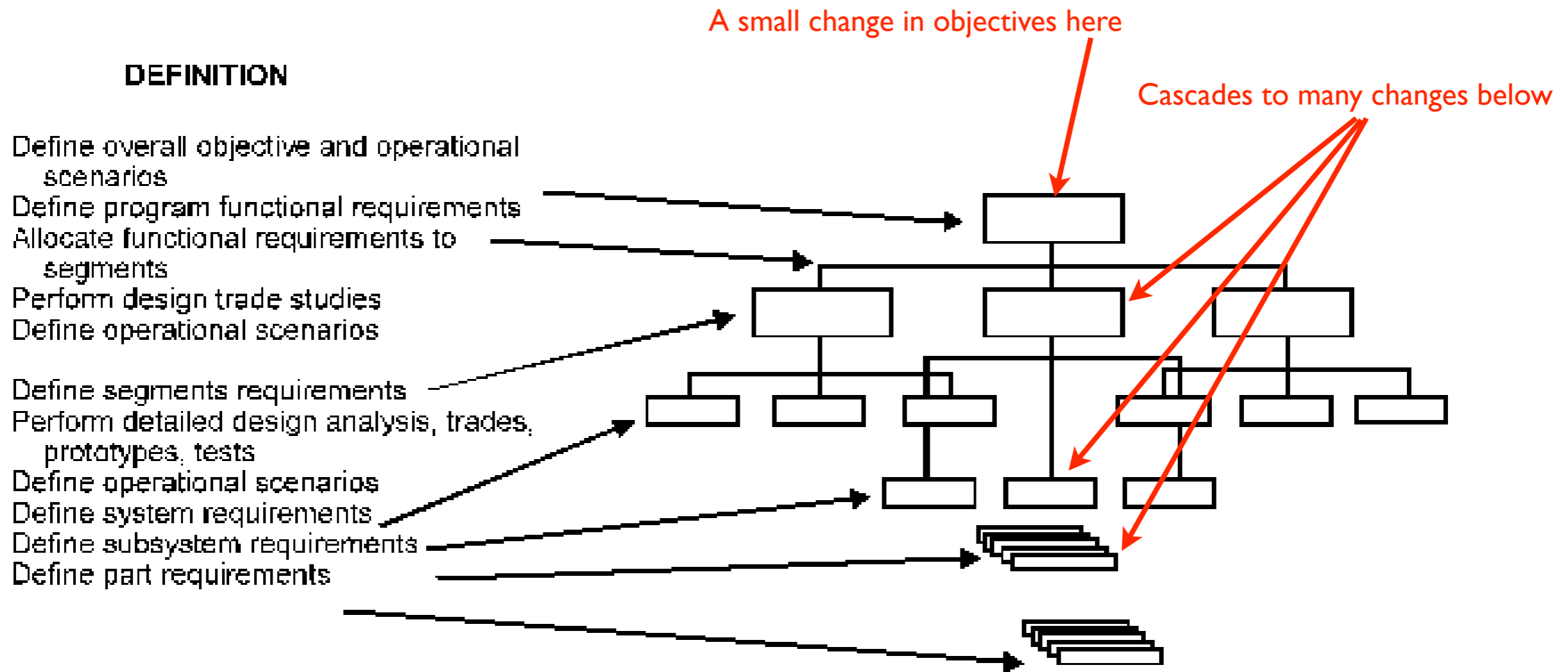


- For each release stage
 1. Generate features to be provided in the product (ex. web access)
 2. Generate requirements these features imply (ex. Apache)
 3. Generate requirements developed from the system design (ex. Server)
 4. Generate requirements developed from the implementation (ex. Hosting)
 5. Generate requirements for the test cases (ex. login, create opera, ...)

Involve All Stakeholders

- Investors
- Marketing
- Management
- Manufacturing
- Testing
- Users

REQUIREMENTS MANAGEMENT PROCESS



<http://www.complianceautomation.com/papers/whyjohnny.htm>

Requirements

- performance
- security
- scalability
- manageability
- maintainability
- reliability
- cost
- risk
- time table
- supportability
- expandability
- power
- air
- light
- heat
- operations
- funding
- staffing
- quality
- legal
- education
- training
- organization
- environmental
- documentation
- interface
- storage
- transmission
- connectivity
- network
- hardware
- software
- apps
- servers
- drivers
- licenses
- support
- and
- so
- on

Decomposition

who, what,
where, when,
why, how

Brainstorm
to generate
more areas

“Do we have
requirements in these
areas?”

Structured walk-through
All aspects of operations
Over the system life-cycle

Good Requirements

Clear, concise, unambiguous
Positive statements
Rationale for it
A need (true business or technical need)
Verifiable (need a process to test it)
Measurable (30mpg on the highway)
Realistic & practical – technical, cost, schedule
Complete
Gathered in a JAD session
Gathered in interviews
Verified after being documented
Flow down to more detailed requirements

“We will land a man on the Moon by the end of this decade”

“I need to lose some weight”

When the rate of change of requirements exceeds the rate of engineering, the project will fail when funds are exhausted

Analyze the Requirements

We will carry only video console titles

Why?

Which titles?

What consoles?

The software handles multiple operas at once

How many?

What size?

What impact?

The software will interface with the opera database

What verification can we do?

Which databases?

The campus will be overbuilt to future-proof it

What about extra space?

Alternatives?

Choose COTS Product

- Create list of must-have and would-be-nice product features in the COTS products
- Eliminate products missing any must-have
- Lab test each candidate COTS product, measure how well each feature is implemented and weight it according to importance. Each product tested results in a weighted score of goodness of fit.
- Pick the top 2 products.
- Do a potential problem analyses and pick the product with the lowest risk

Kepner-Tregoe Decision Analysis

Decision Analysis

Real projects have hundreds of features

		OpenView		Nagios		MRTG	
Feature	Weight	Fit	Score	Fit	Score	Fit	Score
Total sol'n \$	10	50	500	80	800	80	800
Automaps	10	100	1000	70	700	10	100
Autographs	7	60	420	90	540	100	700
Event filters	9	90	810	60	540	10	90
		2,730		2,580		1,690	

MRTG is eliminated because it is the poorest fit

Potential Problem Analysis

		OpenView		Nagios		
Risk	Weight	Fit	Score	Fit	Score	
Licensing \$ rise	10	100	1000	0	0	
Patches not coming	8	100	1000	50	400	
Product Obsoleted	5	60	300	50	250	
Slow innovation	5	80	400	50	250	
			2,700			900

Nagios wins, it has a much lower risk over its lifetime

Pilot Test

- Choose & document core features to be introduced first in this pilot
- Select product champions (stake holders)
- Develop & document requirements to support features
- Develop & document system, data, & project requirements/constraints
- Design a solution around the COTS product previously selected
- Prioritize feature development and assign requirements to developers
- Agree on a test suite for each feature and trace back to each requirement
- Create a project plan for the pilot project (MS Project)
- Go develop and build the pilot solution
- Test the solution, find where it does not pass the test suite
- Go if failures can be fixed on time, Nogo otherwise
- After success, gather lessons learned, new features desired
- Follow with a second, then a final phased implementation

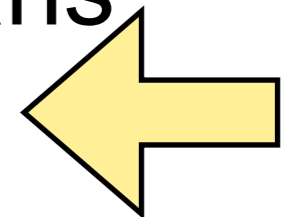
Teams do this analysis

RAD for your Project

- Roll out your product in **3** stages
- Document each stage:
 - Requirements analysis
 - Stages 1 to 6 of the SDLC
 - What is the execution plan
 - Who will you need to participate
 - How much capital is needed

Who, what,
where, when,
why, how

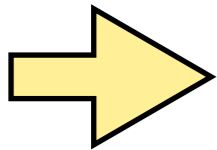
- Discuss the value of presenting these plans in the context of **tools** we don't have yet
- Present the plans, critiques



Process Evaluation

Document these during and after each presentation

- What went well?
- What was hard?
- What was easy?
- How did the lead member perform?
- How did the scribe perform?
- How did the SMEs perform?
- How did the analysts perform?
- What would you do different next time?
- What kinds of **tools** could you have used?



Technical Evaluation

Follow the
SDLC chart

- Is the solution complete?
- Is the solution too complex?
- Did they understand the problem?
- Is the solution correct/accurate?
- How do you know the solution will work?
- Does the solution assume too much?
- Does the solution effect other areas?
- How confident is the team in its results?
- How do you know the solution is feasible?
- What are the risks involved?
- How can these risks be mitigated?

Part 2 - Tools for Requirements Analysis

- Tools discussion
- Diagramming
- Requirement management
- Example requirements
- Open source
- osrmt

Tools

- Tools are no substitute for good skilled people
- You used office productivity tools (OneNote)
- “Powerpoint Engineer”
- What tools did you need in the JAD session?
 - Diagramming tool
 - Requirements Analysis support tool
 - Project Management tool

Diagramming Tools

Helps capture and analyze requirements

- Project Management
- Flow chart
- State diagram
- UML
- Visualization
- Mind Mapping
- Planner (PM)
- KPlato (PM)
- Kivio (Flow chart)
- ArgoUML
- osrmt

Requirements Mgmt

- Authorship (who wrote it)
- Functional (the carton must be stackable 6 high)
- Non-functional (the carton may be brown or grey)
- Project constraints
- Design constraints
- Project drivers
- Project issues
- Fit criteria measures the requirement so you can test the solution against it
- Type (performance, security, data
- Business event
- Importance
- Dependencies
- Conflicts
- Change History

Example Requirement

Requirement #: 75

Requirement Type: 9

Event/use case #: 7, 9

Description: The product shall record all the roads that have been treated

Rationale: To be able to schedule untreated roads and highlight potential danger

Originator: Arnold Snow - Chief Engineer

Fit Criterion: The recorded treated and untreated roads shall agree with the drivers' road treatment logs.

Customer Satisfaction: 3

Customer Dissatisfaction: 5

Priority:

Conflicts:

Supporting Materials:

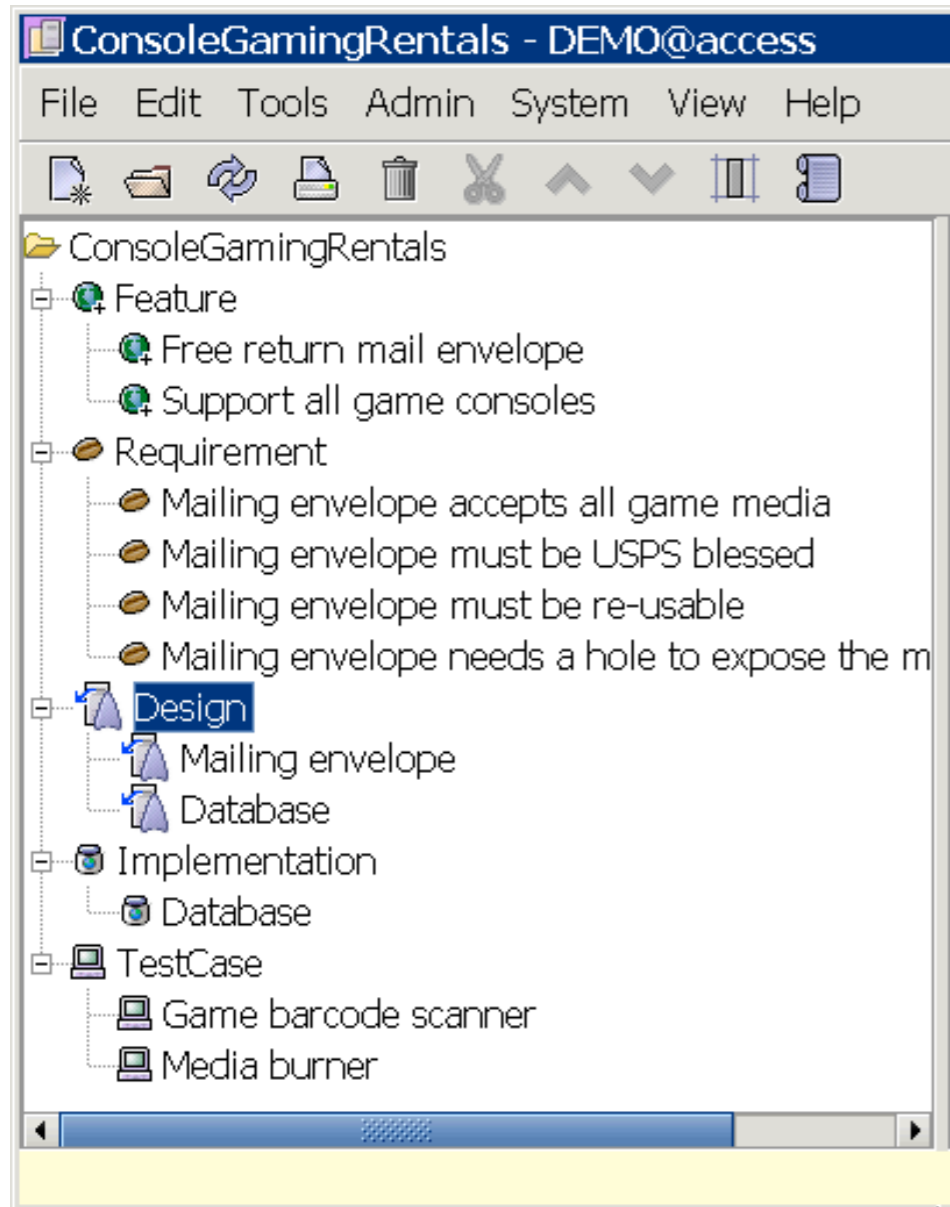
History: Created February 29, 2006

Volere

Copyright © Atlantic Systems Guild

<http://www.volere.co.uk/template.htm>

Open Source



Open Source Requirements Management Tool

“Requirements management tool designed to achieve full SDLC traceability for features, requirements, design, implementation and testing. UI for requirements derivation, version control, common or custom attributes - rationale, source, risk, effort etc”

<http://sourceforge.net/projects/osrmt/>

<http://www.osrmt.com/>

Written in Java

An osrmt Session

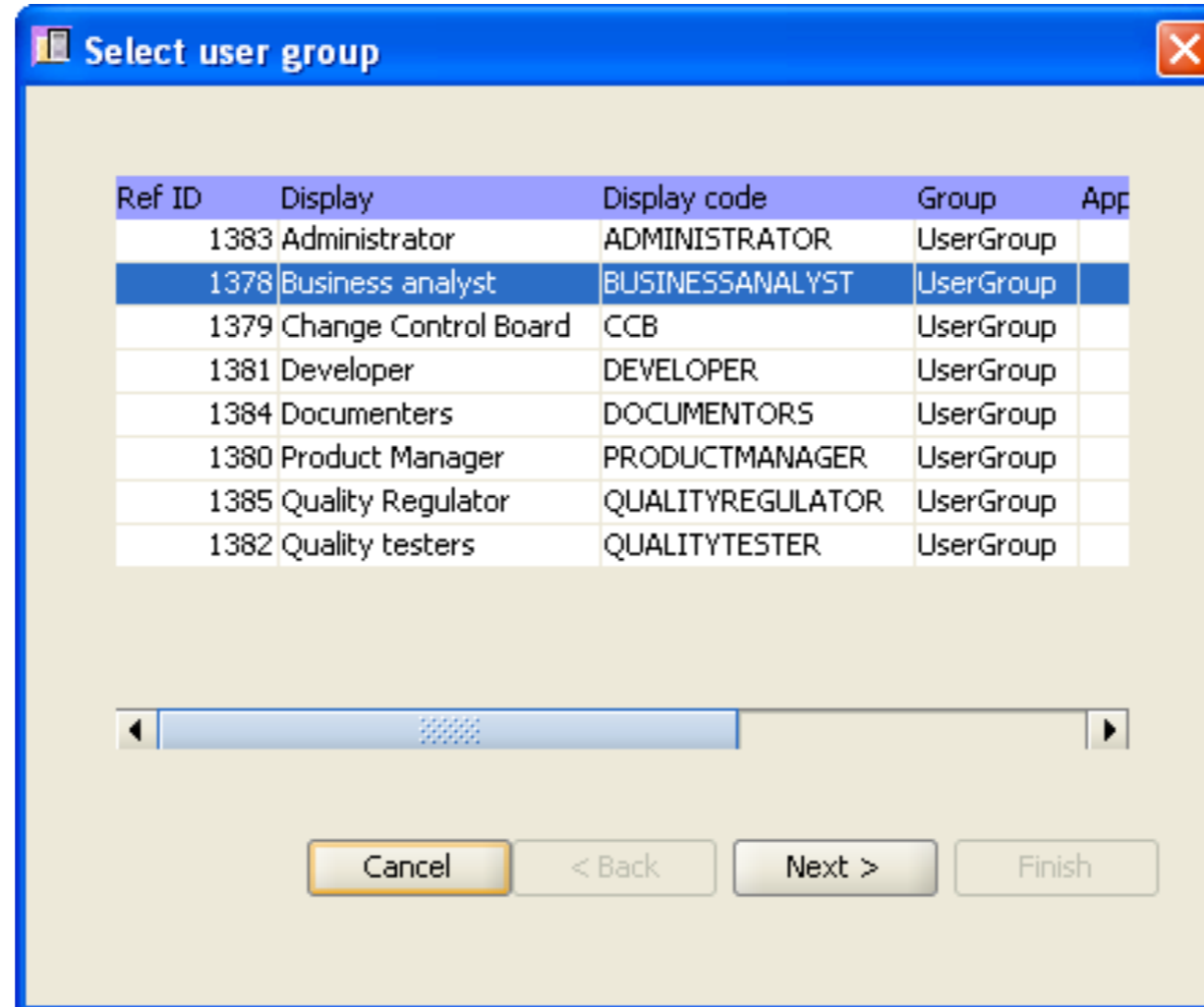
The screenshot shows a software application window titled "ConsoleGamingRentals - DEMO@access". The window has a menu bar with "File", "Edit", "Tools", "Admin", "System", "View", and "Help". Below the menu bar is a toolbar with various icons. On the left side, there is a tree view showing a project structure:

- ConsoleGamingRentals
 - Feature
 - Free return mail envelope
 - Requirement (highlighted)
 - Design
 - Implementation
 - TestCase

The main area of the window displays a table with the following data:

Req #	Name	Priority	Status	Version	Description
1	Mailing envelope accepts all game media	High	Submitted	1.0	The mailing envelope has to acc
2	Mailing envelope must be USPS blessed	High	Submitted	1.0	The USPS defines different shape
3	Mailing envelope must be re-usable	High	Submitted	1.0	The same envelope can be re-use
4	Mailing envelope needs a hole to expose th...	High	Submitted	1.0	The hold exposes the barcode to

User groups



osrmt can run on one computer, or run as a server on one computer and allow remote users to share a requirements database. A web client runs on port 8080.

Requirements, product features, test cases etc.

The screenshot shows a software application window with a menu bar (File, Edit, Tools, Admin, System, View, Help) and a toolbar. The main area is divided into two panes. The left pane shows a tree view of the project structure, with 'Feature' selected. The right pane shows a table of features.

Feature #	Name	Priority	Status	Version	Description
1	System Data Entry	Must have	Completed	1.0	
2	Manual Data Entries	Must have	Completed	1.0	System shall support the manual data
3	Maintain Full Artifact Text	Must have	Completed	1.0	System shall store for editing the full t
4	Binary File Attachments	Must have	Completed	1.0	System shall support the attachment
5	Import Requirements	Important	Completed	1.1	System shall import external requirem
6	Custom Database Fields	Not required	Approved	1.1	System shall allow user definition of a
7	Spellcheck	Not required	Approved	1.1	System shall support spell checking on
8	Externally Linked	Must have	Completed	1.0	System shall support links from the ar
9	Uniquely Identify Artifacts	Must have	Completed	1.1	System shall uniquely identify each ar
10	Define Artifact Hierarchy	Must have	Completed	1.0	System shall support artifacts represe
11	User Defined Fields	Must have	Completed	1.0	System shall support user defined arti
12	System Navigation				
13	Group and Sort Artifacts	Important	Completed	1.1	System shall allow artifacts to be sort
14	Filter List of Artifacts	Important	Completed	1.1	System shall allow the list of artifacts
15	Ad hoc Queries	Important	Submitted	1.1	System shall perform ad hoc queries t
16	Traceability				
17	Identify Source and Origin	Must have	Completed	1.0	System shall be able to identify the so
18	Trace External Artifacts	Important	Submitted	1.1	System shall allow traceability to exte
19	Trace Artifacts	Must have	Completed	1.0	System shall allow maintenance of tra
20	Identify Untraced Requirements	Important	Completed	1.0	System shall identify untraced require
21	Configuration Management				
22	Track Requirement History	Important	Completed	1.0	System shall track entire history of art
23	Version Artifacts	Must have	Completed	1.0	System shall allow for versioning of ar
24	View Related Artifacts	Important	Completed	1.0	System shall allow all related artifacts
25	Change Control Process	Not required	Submitted	1.1	System shall allow for a change contro
26	Baseline artifacts	Must have	Completed	1.1	System shall allow all artifacts to be b

Requirement Form with details, use cases etc.

Requirement [Close]

Requirement | **Details** | Background | Use Case | Relations | History

Goal: Create, retrieve, update and delete hierarchy of artifacts

Context: Artifacts: Features, Requirements, Design, Implementation, Test Cases etc

Precondition: New product created

Main Flow:

Step	
User selects a product	Move Up
User selects an artifact of the desired type	Move Down
User creates a new child artifact	Add Row
System displays a data entry form for the selected type	Remove Row
User enters and saves the new data elements	Apply
System creates a hierarchal relationship between the artifacts	

Alt Flow:

Step	
User updates an existing artifact	Move Up
User moves an artifact to a new artifact of the same type	Move Down
User creates a relationship between two artifacts	Add Row
	Remove Row
	Apply

Postcondition: System saves artifact in hierarchy

Use case goal

OK Cancel

Trace impact of requirement (or any artifact) change

From
the
SDLC

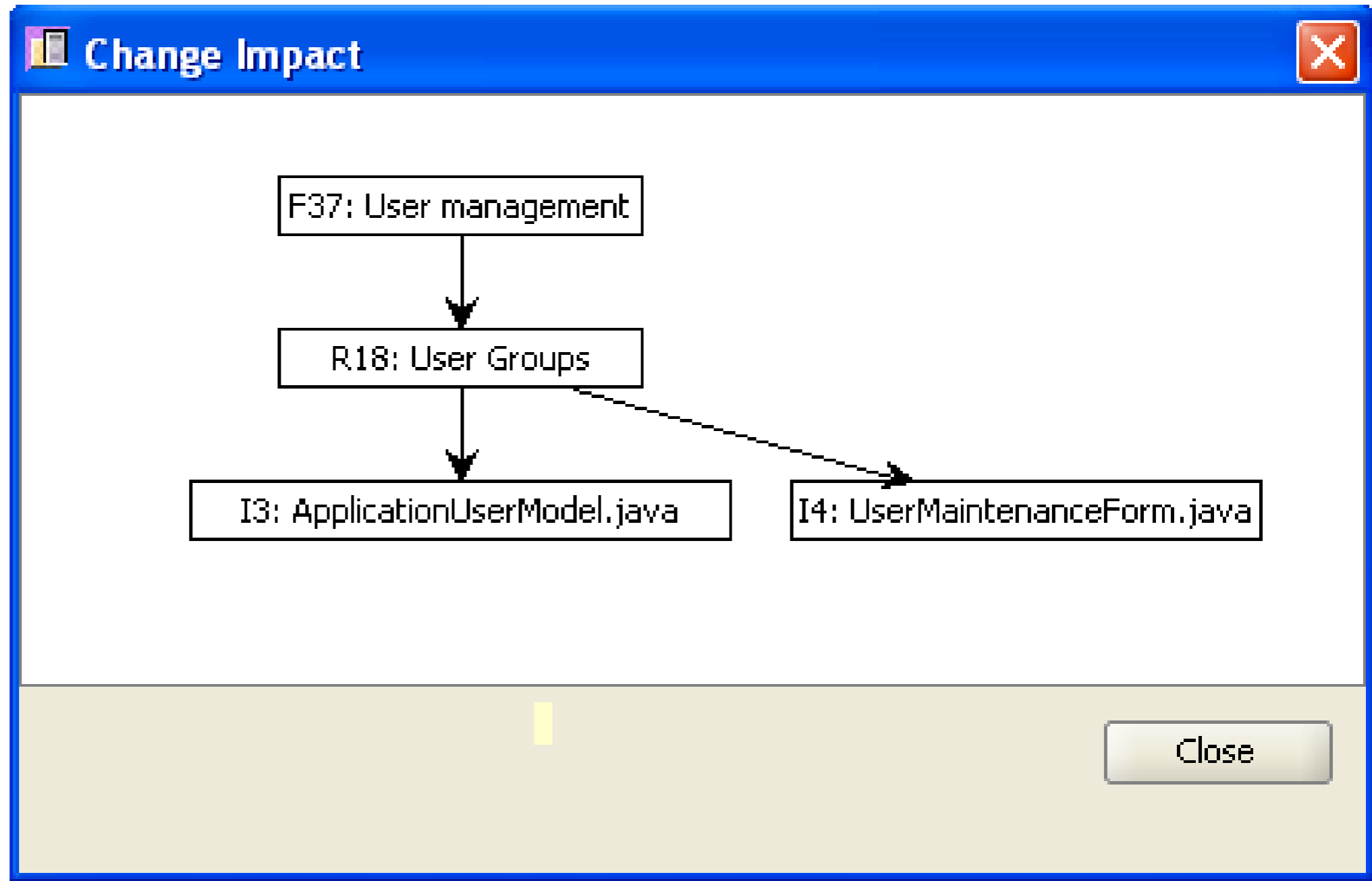
Feature

Requirement

Design

Implementation

Test



Traceability Matrix

Columns: Traceability [X]

Trace From: Trace Type:

Trace To: Apply:

-->	Report param...	Requirement ...	Requirement ...	XML Reports	GUI report wr...	HTML/PDF
Custom Reports				X	X	X
System Log						
Customization						
Define Artifact Hierarchy						
User Defined Fields						
Trace External Artifacts						
Trace Artifacts						
View Related Artifacts						
Change Control Process						
Baseline artifacts						
System Output						
Externally Linked Documents						